1. Analyzing common errors from assignment 3:

**Common mistake 1:** Redeclaring variables in the constructor.

```cpp
class Flight {
private:
    string flight_num;
    string curr_loc;
    string dest;
...
public:
    Flight();
    ...;
};

Flight::Flight () { // constructor
    string flight_num = “A123”;
    string curr_loc = “B”;
    string dest = “A”;
    ...
}
...
```

Analyze the code above. Would the member variables of a Flight object be initialized after calling the constructor? Why or why not? How would you fix the code?
Common mistake 2: Creating extra object(s) when working with class composition.

class Manager {
private:
    Airport* a_arr;
    int num_airports;
    ...
public:
    void populate(); //populate airport(s) detail
    void print_all();
    ...
};

class Airport {
private:
    Flight* f_arr;
    int num_flights;
public:
    void populate_airport();
    void print_airport();
    ...
};

int main () {
    Manager m;
    int num_airports = 3;
    Airport* a_arr = new Airport [num_airports];
    for (int i = 0; i < 3; i++)
        a_arr[i].populate_airport();

    m.print_all();
    ...
}

Assuming all member functions are correctly implemented. Are the Airport objects within the Manager m loaded/populated? Why or why not? How would you fix the code?
**Common mistake 3**: A chain of accessor calls.

```cpp
class Manager {
    void print_virtual_first_flight() {
        cout << "Flight num: " << a_arr[0].get_f_arr()[0].get_flight_num() << endl;
        cout << "Current at: " << a_arr[0].get_f_arr()[0].get_curr_loc() << endl;
        cout << "Destination: " << a_arr[0].get_f_arr()[0].get_dest() << endl;
    }
};
```

In `main()`:
Manager m;
```
m.print_virtual_first_flight();
```

What is the issue with the `print_virtual_first_flight()` function above? Why is it a bad idea to use a chain of accessors to get the internals of `Flight` class from the `Manager`? How would you fix the code?

**Class Relationship:**
2. Given the following possible classes, list at least three “has-a” relationship, and three “is-a” relationship.

```
Animal       Dog       Shape       Mammal       Triangle       Teeth
Vehicle      Person    Driver      Wheel       Truck         Space Shuttle
```
Accessibility:
3. Explain the difference between public, private, and protected.

Inheritance:
Given the following code, discuss the following and write code to prove your answers.

```cpp
struct Card {
    int rank;       // 1-13
    string suit;    // “heart”, “spade”, “diamond”, “club”
};

class Cardgame {
    protected:
        Card *deck;
        int num_cards;
    public:
        Cardgame();
        ~Cardgame();
        void play_game();
};

class Gofish : public Cardgame {     // inherited from Cardgame
    private:
        int max_players;
    public:
        Gofish();
        ~Gofish();
};
...
```

4. If we create a child object, i.e. Gofish g;
   a. What is inherited and not inherited?

   b. What is accessible and not accessible?

   c. In what order is the Cardgame constructor and Gofish constructor called?

   d. When the object g is out of scope, in what order is the Cardgame destructor and Gofish destructor called?